Security Verification of Key Exchange in Ciphertext-Policy Attribute Based Encryption

Baasansuren Bat-Erdene
Faculty of Informatics
Eötvös Loránd University
Budapest, Hungary
heqeyo@inf.elte.hu

Yuping Yan
Faculty of Informatics
Eötvös Loránd University
Budapest, Hungary
yupingyan@inf.elte.hu

Mohammed B. M. Kamel
Eötvös Loránd University
Budapest, Hungary
Furtwangen University
Furtwangen, Germany

Peter Ligeti
Faculty of Informatics
Eötvös Loránd University
Budapest, Hungary
ligetipeter@inf.elte.hu

2022 7th International Conference on Signal and Image Processing (ICSIP) | 978-1-6654-9563-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICSIP55141.2022.9887218

***Abstract*—Attribute-based encryption (ABE) is an extension scheme of identity-based encryption and public-key encryption. It can achieve fine-grained access control and one-to-many encryption mode, suitable for practical applications. In addition to mathematical proofs, it is essential to verify its security properties using different protocols. This paper proposes two key exchange protocols to securely exchange the user secret keys in ABE. ProVerif is an automated cryptographic protocol verifier that we have used during protocol verification. We specifically aim to protect the confidentiality of the generated keys. The proposed protocols are formally analysed and their security property has been formally proved.**

***Index Terms*—Attribute Based Encryption, Ciphertext-Policy ABE, Key exchange confidentiality, Formal verification**

I. Introduction

A user can access the data in some distributed systems if only that user has been provided with specific permissions. To provide such permissions, trusted server can be used to store data and corresponding access control [1]. Using the Attribute Based Encryption (ABE), the encrypted data can be kept confidential even if storage is untrusted.

ABE is an extension scheme of Identity Based Encryption (IBE) and public-key encryption. Compared to the traditional asymmetric encryption schemes, the ABE can effectively support one-to-many encryption mode, which is more suitable in the distributed environments with multiple receivers [2]. With implementing various threshold and logic operations, ABE develops more flexible access control strategies to control the users' privileges. Due to these significant advantages, ABE has an increasing applications in the fields of distributed data management, third-party data storage, directional broadcast, and so on [3].

As ABE is popularly implemented, it is essential to study key exchange protocols of the generated keys in ABE to guarantee the confidentiality of the key exchange process [4]. In this paper, we use ProVerif automatic cryptographic tool to verify the confidentiality property during the key exchange of the ABE generated secret keys in various protocols.

Outline. In Section 2, we discuss some of the related efforts in the field of ABE. We state the primitives of CP-ABE, the ProVerif tool, key distribution center, and public key infrastructure in Section 3. Using the ProVerif tool, in Section 4, we formally model our proposed protocols for key exchange in CP-ABE and analyse the confidentiality security property. Finally, we conclude the results in Section 5.

II. Literature Review

Amit Sahai and Brent Waters [5] introduced a new type of IBE scheme called Fuzzy Identity-based encryption. In traditional public-key cryptography, a message is encrypted with the receiver's public key, whereas in ABE, a message is encrypted with a set of attributes, and the message can be decrypted by any receiver who has a set of keys for matching attributes. There are two types of ABE: key-policy attribute-based encryption (KP-ABE) and ciphertext-policy attribute-based encryption (CP-ABE). These two versions differ in the phase where the access policies are settled.

In KP-ABE [6], the ciphertext is associated with a set of attributes, and the secret key is associated with the access tree. The sender does not define the privacy policy and has no control over who has access to the data except by defining the set of specific attributes that are necessary to decrypt the ciphertext. The Authority that generates the user's secret key establishes the combination of attributes for which the private key can be used. In CP-ABE, the idea is reversed. The secret keys are associated with a set of attributes, the ciphertext is associated with the access tree, and the sender can determine the access policy under which the data can be decrypted [3].

ProVerif can be utilized to detect symbolic attacks in a protocol. Yap et al. [7] Studied the proof of correctness of MQV-based key exchange (AKE) protocol using ProVerif, as well as some of the known computational attacks done by others such as Unknown-Key-Share attack using it. Their results included identity based key agreement and certificate-less identity authenticated based key agreement variants of MQV-based protocol. Edris et al. [8] proposed a device to

This research was partially supported by Application Domain Specific Highly Reliable IT Solutions project which has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the Thematic Excellence Programme TKP2020-NKA-06 (National Challenges Subprogramme) funding scheme, Project no. TKP2021-NVA-29 has been implemented with the support provided by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021-NVA funding scheme and SH programme.
Corresponding Author: mkamel@{inf.elte.hu, hs-furtwangen.de}

Authorized licensed use limited to: Universitaet Bielefeld. Downloaded on October 27,2023 at 07:55:11 UTC from IEEE Xplore. Restrictions apply.

device solution that comprises D2D Service security (DDSec) and Capability security (DDACap) security protocols. Their proposal provides security for access, caching and sharing data in network-assisted and non-network-assisted device to device communications scenarios. They proved the security of their protocol using ProVerif.

Authors in [4] used the ProVerif to discuss the confidentiality of the CP-ABE secret keys. Rajeb et al. [9] formally verified the cloud storage protocol based on Attribute Based Signature (ABS) and ABE schemes that is proposed by Ruj et al. [10], using ProVerif. They discovered an unknown attack against user privacy, and proposed a correction. They automatically proved the security of the their corrected protocol using ProVerif.

III. Primitives

A. CP-ABE

In CP-ABE, a user's private key is associated with a set of attributes expressed as strings, and ciphertext is specified by an access policy over attributes. A user will be able to decrypt a ciphertext if that user's attributes satisfy the policy of the corresponding ciphertext [3]. Access policies can be defined over attributes using $\wedge, \vee$, n-of-n threshold gates, and 1-of-n threshold gates. For example, let us assume we have a set of attributes $\{$A,B,C,D$\}$ and *User1* has the keys for attribute $\{$A,B$\}$, while *User2* has the key for attribute $\{$D$\}$. If a sender encrypts the message with (A$\wedge$C)$\vee$D access policy, then only *User2* is able to decrypt the message and *User1* is not able to decrypt it [3].

CP-ABE scheme consists of four main algorithms, including Setup, Encrypt, KeyGen, and Decrypt [3].

- **Setup.** The setup algorithm takes as input the security parameter and produces the public parameters PK and a master key MK.
- **Encrypt.** The encryption algorithm takes as input the public parameters PK, a message M, and an access structure A. The algorithm encrypts message M and produces a ciphertext CT as output. The ciphertext CT, which contains an access policy A, can be decrypted only if a user has attributes that match the access structure.
- **KeyGen.** The key generation algorithm takes as input the master key MK and a set of attributes S which describe the key. It produces a private key SK as output, a private key for a set S of attributes.
- **Decrypt.** The decryption algorithm takes as input the public parameters PK, a ciphertext CT, and a private key SK. If the set S of attributes satisfies the access structure A, the algorithm will decrypt the ciphertext and return a message M.

Figure 1 illustrates the main structure of CP-ABE.

1) In the Authority part, the Setup algorithm is worked and got public key PK and master key MK. Then it sends a public key to the sender.
2) Sender runs Encrypt algorithm to get ciphertext C. In this part, the sender determines access policy A_{c-cp} on who can decrypt. Only (AI $\vee$ Cybersecurity) $\wedge$ (BSc $\vee$ MSc) students can see the data in the following example.
3) Every user has its attributes. They send their attributes to Authority, and then the key generation algorithm is worked in the Authority part to output the secret key. After getting the private key, a receiver can decrypt the message if that user's attribute matches the access policy.

One of the main challenges in the described steps that this paper addresses is how the generated secret keys are sent to the corresponding user securely.

B. ProVerif

ProVerif [11] is an automatic cryptographic protocol verifier, which has been used and developed since 2001. It can formally provide a proof for various security properties. ProVerif takes a protocol model that is expressed in an extension of the pi-calculus as inputs. Additionally, it takes the security properties that have to be formally proved. Then, it automatically translates this information into an internal representation and uses an algorithm based on resolution with free selection to determine whether a fact is derivable from the clauses or not.

C. KDC

The Key Distribution Center (KDC) is a server trusted by all users and shares a secret key with each user. In KDC, the Key Encryption Key (KEK) is used to securely transmit session key. Using KDC, each user shares a unique secret key KEK with the key distribution center, which is pre-distributed through a secret channel. When Alice wants to connect with Bob, KDC encrypts the session key used in communication between Alice and Bob. In the basic protocol, KDC creates two messages for Alice and Bob respectively $Y_A = Enc_{KA}(K_{ses}), Y_B = Enc_{KB}(K_{ses})$. After getting their session key, Alice and Bob can securely communicate with each other.

D. PKI

Public-key cryptography uses public and private key pairs to encrypt and decrypt the message between users. A user who intends to communicate securely distributes its public key and keeps its private key secret. Public Key Infrastructure is the set of hardware, software, policies, processes, and procedures required to create, manage, distribute, use, store, and revoke digital certificates and public-keys. It helps to address the key distribution of system participants and provides a mechanism to valuate the distributed public keys.

IV. Protocol Verification

We verified that our proposed protocol guarantees confidentiality property during the key exchange process of CP-ABE. For the secret data in our protocol model, we define a secret variable $pMsg$. The secret value is then used as the plaintext for the encrypted messages sent by the parties. Without proper security, an active attacker is able to retrieve a message's plaintext $pMsg$ using the query: query attacker(pMsg).

Authorized licensed use limited to: Universitaet Bielefeld. Downloaded on October 27,2023 at 07:55:11 UTC from IEEE Xplore. Restrictions apply.

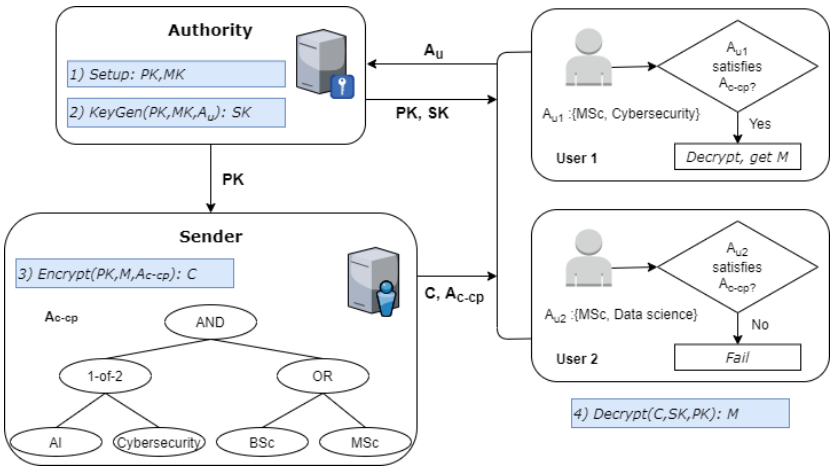

Fig. 1. Ciphertext-Policy Attribute-based Encryption

During our implementation of CP-ABE in ProVerif, without further security consideration, an eavesdropper can intrude and decrypt the transmitted message, by having access to the exchanged CP-ABE secret key. Our proposed protocols aim to achieve the key exchange confidentiality property in CP-ABE, utilizing PKI [12] and KDC technologies. In the following, each of the two proposed protocols is stated.

A. CP-ABE modeling

Our model in ProVerif consists of six main parts: defining variables and functions, defining ABE scheme, main process, encryption process, decryption process, and attribute authority process. Detailed CP-ABE flow diagram is illustrated in Figure 2.

Defining ABE scheme. The modelling of ABE scheme is shown in Code 1. The constructor $abePk$ takes abeKey as input and outputs $abeKey$, the public key parameter for ABE. The constructor $abeSka$ takes as input $abeKey$ and a set of attributes, and outputs secret key. The constructor $abeEnc$ takes as input $abeKey$, the message, and the access policy, then outputs ciphertext. The constructor $abeDec$ takes as input the ciphertext, and secret key then outputs the message. The constructor $abeCheckKey$ checks whether the user's secret key and set of attributes match the private key.

```
fun abePk(abeKey):abeKey.
fun abeSka(abeKey,bitstring,bitstring):bitstring.
fun abeEnc(abeKey,bitstring,bitstring):bitstring.
fun abeDec(bitstring,bitstring):bitstring.
fun abeEval(bitstring,bitstring,bitstring):bitstring.
fun abeCheckKey(abeKey,bitstring,bitstring,bitstring)
    :bitstring.
    equation forall  uid:bitstring,
    msg:bitstring, sk1:abeKey,
    ska1:bitstring, att:bitstring;
abeDec(abeEnc( abePk(sk1), AccessPolicy, msg),
        abeSka(sk1,uid,att) )=
abeEval(AccessPolicy, msg,
abeCheckKey(abePk(sk1),abeSka(sk1,uid,att),uid,att));
forall  sk1:abeKey, uid:bitstring,att:bitstring;
abeCheckKey(abePk(sk1),abeSka(sk1,uid,att),uid,att)
    =pass;
forall  msg:bitstring;
    abeEval(AccessPolicy,msg,pass)=msg.
```

Listing 1. ABE scheme

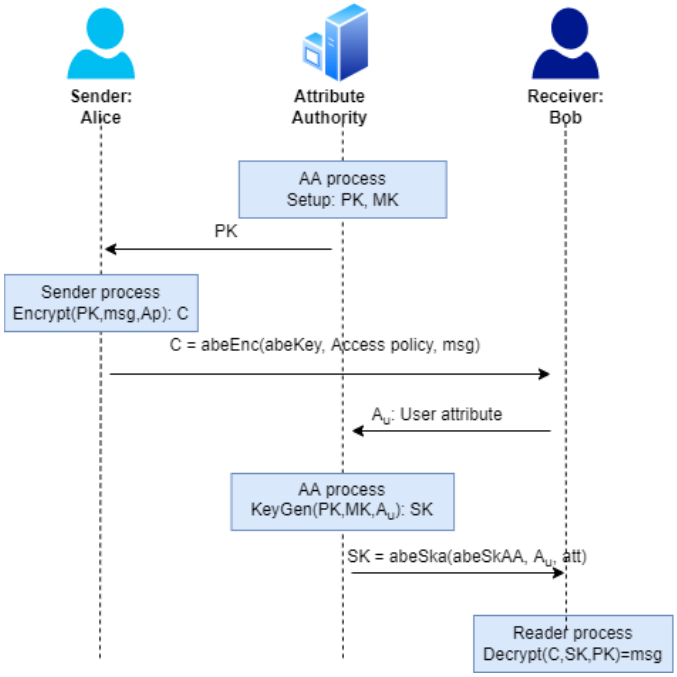

Fig. 2. CP-ABE overview

Encryption process. The role of a sender is modeled by the Encryption process given in Code 2. The sender's key and attribute authority keys are established. Then sender encrypts the secret message with the public key of the attribute authority and sends it through the public channel.

```
let Sender(pkSender:pkey, abePkAA:abeKey)=
let abeEncMsg = abeEnc(abePkAA, AccessPolicy, pMsg ) in
out( c, abeEncMsg ).
```

Listing 2. Encryption process

Decryption process. The role of a receiver is modeled by the Decryption process specified in Code 3. The receiver first gets his private key from the attribute authority, and then he

Authorized licensed use limited to: Universitaet Bielefeld. Downloaded on October 27,2023 at 07:55:11 UTC from IEEE Xplore. Restrictions apply.

decrypts the message using his attribute secret key $abeSkAtt1$ and behaves as *event e* with the received message.

```
let Receiver(skReceiver:skey, pkReceiver:pkey,
abePkAA:abeKey)=
in( c, abeEncMsg:bitstring ) ;
out( c, userid(pkReceiver));
in( c , abeSkAtt1:bitstring );
in( c , encAbeSkAtt1:bitstring );
let abeSkAtt1 = adec(skReceiver, encAbeSkAtt1)in
if abeCheckKey(abePkAA,abeSkAtt1,userid(pkReceiver),
att1) = pass
    then let m = abeDec(abeEncMsg, abeSkAtt1) in
event e.
```

Listing 3. Decryption process

The Attribute Authority (AA) process. The Attribute authority process is given in Code 4. AA generates the private key and sends it to the user.

```
let AA(pkReceiver:pkey, abeSkAA:abeKey)=
in(c, userX:bitstring) ;
let userA = userX in
if userA=userid(pkReceiver) then
out( c, abeSka(abeSkAA, userA, att1) ).
```

Listing 4. Attribute authority process

The main process. The main process is given in Code 5. First, new secret key $skSender$ for sender and $skReceiver$ for receiver are generated. Then, their corresponding public keys are distributed through the public channel, making them available to all system members, including the attacker. Additionally, a new secret key $abeSkAA$, which is the secret ABE key used by the attribute authority, is generated, and the corresponding public key is distributed through the public channel. Finally, the key generation, encryption and decryption are executed.

```
process
 new skSender:skey ; new skReceiver:skey;
 let pkSender=pk(skSender) in out( c, pkSender) ;
 let pkReceiver=pk(skReceiver) in out( c, pkReceiver) ;
 new abeSkAA:abeKey ; let abePkAA=abePk(abeSkAA) in
 out( c,abePkAA ) ;
Sender(pkSender,abePkAA)|Receiver(skReceiver, pkReceiver,
abePkAA)
|AA(pkReceiver, abeSkAA)
```

Listing 5. Main process

B. Using KDC in CP-ABE.

As illustrated in Figure 3, the participating parties in the proposed protocols are the sender, Alice, the receiver, Bob, the Attribute Authority, and KDC. The basic steps are following:

1) Setup and KeyGen algorithms run in Attribute Authority. The Attribute Authority sends the Public key through the public channel. Among all participating parties, the Alice receives the ABE public key of the attribute authority.
2) Alice encrypts the secret data using CP-ABE encryption algorithm and the received ABE public key of the attribute authority and sends it to Bob.
3) Bob presents his identity to the Attribute Authority.
4) After verifying Bob's identity and his attributes, the Attribute authority generates a secret key and sends a connection request to KDC.

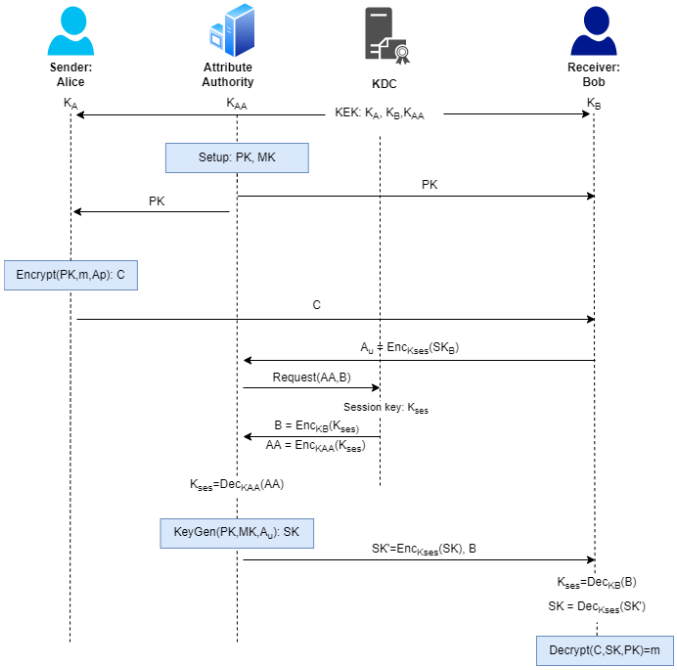

Fig. 3. Using KDC in CP-ABE

5) KDC generates a session key and sends it encrypted with their public key.
6) The Attribute Authority sends a secret key encrypted with the session key to Bob.
7) Bob receives the ciphertext and his encrypted secret key. After decrypting his ABE secret key using the session key, and by using the already received public key of the Attribute Authority, he decrypt the ciphertext.

C. Using PKI in CP-ABE.

As illustrated in Figure 4, the participating parties are the sender, Alice, the receiver, Bob, the Attribute Authority, and Certificate Authority (CA). The basic steps are following:

1) Setup and KeyGen algorithms run in Attribute Authority. The Attribute Authority sends the Public key through the public channel. Among all participating parties, the Alice receives the ABE public key of the attribute authority.
2) Alice encrypts the secret data using the received public key of the attribute authority and sends it to Bob, the receiver through public channel.
3) Bob sends a request to get a secret key to Attribute Authority. During the negotiation, Bob sends his signed certificate to the Attribute Authority.
4) After verifying Bob's identity and his attributes, Attribute Authority sends the secret key signed with its secret key and encrypted with Bob's public key.
5) Bob verifies the signature by using the signed certificate by the CA that includes the attribute authority public key.

Authorized licensed use limited to: Universitaet Bielefeld. Downloaded on October 27,2023 at 07:55:11 UTC from IEEE Xplore. Restrictions apply.

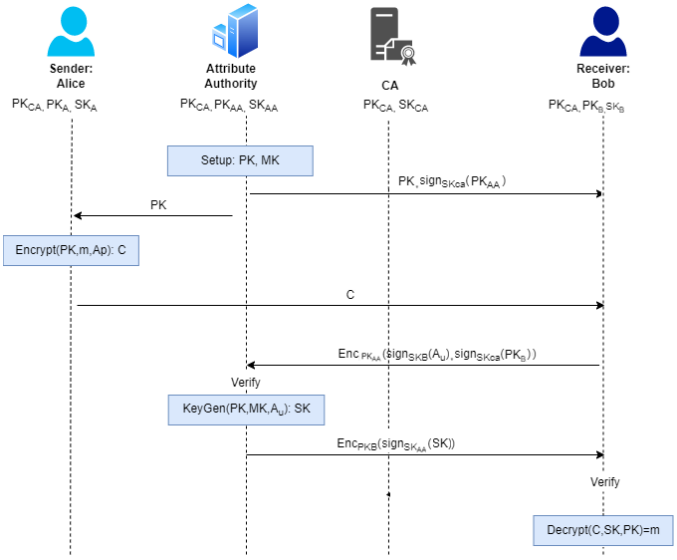

Fig. 4. Using PKI in CP-ABE

```
Verification summary:

Query not attacker(pMsg[]) is true.

Non-interference pMsg is true.
```

Fig. 5. Verification summary of the proposed protocols

6) After verifying and decrypting the received secret key, Bob is able to decrypt the transmitted secret data by Alice.

D. Analysis

Our proposed protocols have been implemented in ProVerif, and formally verified to prove the confidentiality property of the ABE secret key during the key exchange process between the attribute authority and the user. As it is shown in Figure 5, this can be proved by querying an attacker's ability to access the transmitted private message, as a consequence of having access to the ABE secret key.

Confidentiality means that a user can access the encrypted data if and only if the user satisfies the requirements of access control policies. In ProVerif, this property can be expressed as secrecy property by verifying the inability of the attacker to access the private data. In CP-ABE protocol implementation, without further consideration of the key exchange, an attacker can get the encrypted message, using the transmitted ABE secret key. To address this issue, two protocols have been proposed to protect the secrecy of the secret keys.

- Using KDC that is a trusted server by all users and shares a secret key with each user. Using KDC in CP-ABE, the secret key is sent after being encrypted with KEK. ProVerif proves secrecy property as shown in Figure 5.
- Using PKI in CP-ABE, the secret key is sent after being encrypted by the public key of the receiver that is already signed by the secret key of the CA entity. ProVerif proves secrecy property as shown in Figure 5.

Proposition 1. *Using KDC and PKI in CP-ABE secret key exchange, the confidentiality property is guaranteed.*

V. Conclusion

In this paper, we verified the confidentiality property of CP-ABE secret key exchange in two different protocols using ProVerif. We showed that an adversary could intrude and decrypt the ciphertext without a proper protocol for signing and encrypting the user's secret key. The first proposed protocol is based on the Key Distribution Center (KDC), while the second protocol is based on the use of Public Key Infrastructure (PKI) as an infrastructure with trusted CA entity to check the public key and send the corresponding keys for signing and encryption. The results of verifying the proposed protocols in ProVerif showed that these two protocols can achieve the confidentiality property, and protect the transmitted ABE secret keys during the key exchange process.

References

[1] M. Kamel, Y. Yan, P. Ligeti, and C. Reich, "Attred: Attribute based resource discovery for iot," *Sensors*, vol. 21, no. 14, p. 4721, 2021.

[2] Y. Yan, M. B. Kamel, and P. Ligeti, "Attribute-based encryption in cloud computing environment," in *2020 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, pp. 63–68, IEEE, 2020.

[3] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE symposium on security and privacy (SP'07)*, pp. 321–334, IEEE, 2007.

[4] B. Bat-Erdene, Y. Yan, and M. B. Kamel, "Formal verification of confidentiality in attribute-based encryption through proverif," in *Book of Abstracts, 21th Central European Conference on Cryptology*, p. 34, 2021.

[5] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Annual international conference on the theory and applications of cryptographic techniques*, pp. 457–473, Springer, 2005.

[6] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, pp. 89–98, 2006.

[7] E.-Y. Yap, J.-J. Chin, and A. Goh, "Verifying mqv-based protocols using proverif," in *IT Convergence and Security*, pp. 55–63, Springer, 2021.

[8] E. K. K. Edris, M. Aiash, and J. Loo, "Formal verification of authentication and service authorization protocols in 5g-enabled device-to-device communications using proverif," *Electronics*, vol. 10, no. 13, p. 1608, 2021.

[9] M. Berrima, P. Lafourcade, M. Giraud, and N. B. Rajeb, "Formal analyze of a private access control protocol to a cloud storage.," *SECRYPT*, vol. 2017, p. 14th, 2017.

[10] S. Ruj, M. Stojmenovic, and A. Nayak, "Privacy preserving access control with authentication for securing data in clouds," in *2012 12th IEEE/ACM International symposium on cluster, cloud and grid computing (ccgrid 2012)*, pp. 556–563, IEEE, 2012.

[11] B. Blanchet, B. Smyth, V. Cheval, and M. Sylvestre, "Proverif 2.02 pl1: Automatic cryptographic protocol verifier, user manual and tutorial," 2020.

[12] S. Choudhury, K. Bhatnagar, and W. Haque, *Public key infrastructure implementation and design*. John Wiley & Sons, Inc., 2002.

Authorized licensed use limited to: Universitaet Bielefeld. Downloaded on October 27,2023 at 07:55:11 UTC from IEEE Xplore. Restrictions apply.